DOCUMENT-IDENTIFIER: US 6125383 A
TITLE: Research system using multi-platform object oriented program language for providing objects at runtime for creating and manipulating biological or chemical data

BSPR:
Although Web (i.e., network) technology allows users at a desktop computer to access programs and databases on remote computers, such programs lack a unifying standard. In particular, these programs have their own unique interface--program specific format for input and output. Ease of use is sacrificed, since users must learn to operate many different programs and must jump formidable technical hurdles to exchange data between these programs. As this often involves laborious and tedious manipulation of data files as well as
detailed knowledge of the operations of programs and the quirks of each operating system, the chances of **error** are significant. Currently, scientists either spend unnecessary hours to accomplish tasks with these tools, or simply choose not to try, and potentially miss important observations.

DEPR:
The present invention incorporation of Java and CORBA provides for organizing and integrating bioinformatics and/or chemiinformatics data sources and analytical engines. Analysis tools and databases are integrated into the system through the creation of "**wrappers**". The creation of "**wrappers**" is well known to those skilled in the art, and therefore discussion to such is eliminated for sake of brevity. In short, a **wrapper** is a small program that encapsulates the knowledge of each server program's requirements, its inputs and output formats, and its quirks. Unlike conventional systems, the present invention frees the user from the burden of learning how to operate many different programs. Since the inputs and outputs of the **wrapper** are in a common format, all of the data sources and analytical engines available on substantially any server, including legacy or preexisting systems, can be made to intercommunicate.

DEPR:
FIG. 8 is a drawing depicting a network architecture 300 according to one aspect of the present invention. It will be appreciated that various architectures may be employed to carry out the present invention, all which fall within the scope of the present invention. The architecture 300 consists of several network layers. The architecture 300 is divided up to show the architecture on the client side 310 and the architecture on the server side 320. The bottom layer 326 is the transport protocol layer. In the preferred embodiment, the transport protocol layer 326 is a TCP/IP layer--of course any suitable type of transport protocol layer may be employed to carry out the present invention. The transport protocol layer 326 exists on both the client side 310 and the server side 320. This layer 326 in general serves to transport data without **error** or loss.

DEPR:
On top of the transport layer 326 is a marshaling protocol layer 330 on both the client side 310 and server side 320. In this embodiment, the marshaling protocol layer 310 is a IIOP ORB (Internet Interoperation Protocol Object Request Broker). The marshaling protocol layer 320 provides for connecting client objects with **server objects**. On the client side 310, over the marshaling protocol layer 330 is a client stub 332 which provides for gluing the project client 336 to the marshaling protocol layer 330, and it also

provides for connecting the project client 336 to the server stub 340. The project client 336 serves to manage projects on the client side 310 for a particular client 110. The project client's 336 preferred implementation is Java, however, it should be appreciated that other programming languages could be used for the proj. client 336. In general, the proj. client 336 is a cached copy of a corresponding team project stored on the server 112. On top of the proj. client 336 is a client view layer 350 which provides for viewing data and analysis in general.

DEPR:
However, the present invention provides for accomplishing the above with a two-step process. In a similar manner, the client 110 is capable of adding and
manipulating specific DNA sequence data, by calling services on the DNA object,
with assistance from the server 112. For example, if the client 110 wants to compare DNA sequence with a database, to see if there are similar DNA sequences
which have been identified, using a service called BLAST, the **client would invoke** a method on the DNA sequence object to call the service object "BLAST" on the server 112. This is, again, accomplished through the instantiation of a
proxy object "DNA Sequence.ADH.sub.-- Human.BLAST". The proxy object is again aware of where in the server 112 the service for BLAST is located, and carries with it the parameters (arguments) for the particular DNA object it is serving.

DEPR:
Objects that can be annotated or merged such as a biology or chemical research document may be opened by all researchers for viewing. However, the view of each user with respect to the research document may be different. As a result several users may access and write to a document simultaneously since no two users have both read and write access to the same portion of the document. Thus, large scale research and collaborative efforts is facilitated by allowing
many people on a research team to view and possibly manipulate an object such as a DNA sequence and then together produce a research paper at high speed with
a significant reduction in **errors** since no user can overwrite or modify a portion of a document that is assigned to another.

DEPR:
Through the use of **wrappers**, as described above, input and output from various programs are always converted to a common format. This allows the results from
one analysis to be automatically used as input for further analyses (e.g., the results from a database search can be automatically entered into a multiple sequence alignment).